Cellular automata on regular rooted trees

Tullio Ceccherini-Silberstein[1], Michel Coornaert[2], Francesca Fiorenzi[3], and
Zoran Šunić[4]

[1] Dipartimento di Ingegneria, Università del Sannio
C.so Garibaldi 107, 82100 Benevento, Italy
`tceccher@mat.uniroma3.it`
[2] Institut de Recherche Mathématique Avancée, UMR 7501, Univ. Strasbourg
7 rue René Descartes, 67084 Strasbourg Cedex, France
`coornaert@math.unistra.fr`
[3] Laboratoire de Recherche en Informatique, UMR 8623, Bât 650 Univ. Paris-Sud 11
91405 Orsay Cedex France 91405 Orsay, France
`fiorenzi@lri.fr`
[4] Department of Mathematics, Texas A&M University
MS-3368, College Station, TX 77843-3368, USA
`sunic@math.tamu.edu`

Abstract. We study cellular automata on regular rooted trees. This includes the characterization of sofic tree shifts in terms of unrestricted Rabin automata and the decidability of the surjectivity problem for cellular automata between sofic tree shifts.

Keywords: Free monoid, sofic tree shift, unrestricted Rabin automaton, finite tree automaton, cellular automaton, surjectivity problem.

1 Introduction

In this paper, we study cellular automata between subshifts of A^{Σ^*} (also called tree shifts), where A is a finite nonempty set and Σ^* is a finitely generated free monoid identified with the $|\Sigma|$-regular rooted tree. We investigate, in particular, the decidability of the surjectivity problem for these cellular automata.

Amoroso and Patt [1] proved that the surjectivity and the injectivity problems have a positive answer in the one-dimensional case (i.e. when Σ^* is replaced by $\mathbb{Z}$ or $\mathbb{N}$). On the other hand, Kari [8] proved that these problems fail to be decidable in dimension $d \geq 2$ (i.e. for cellular automata defined on $A^{\mathbb{Z}^d}$). There are more general algorithms to decide the surjectivity of a cellular automaton on a finitely generated free monoid (which are deducible combining the results of Rabin [11], Muller and Schupp [10] and Thatcher and Wright [12]), but they are not practical. We have worked out the details in a limited setting of interest, namely when we start with an unrestricted Rabin automaton.

Tree shifts have been extensively studied by Aubrun and Béal in [2], [3] and [4]. In the present work we use a slightly different (but equivalent) setting.

A tree shift is said to be of finite type if it can be described as the set of configurations avoiding a finite number of forbidden patterns. Sofic tree shifts

are defined as the images of tree shifts of finite type under cellular automata. In the one-dimensional case, a sofic subshift of $A^{\mathbb{N}}$ may be characterized as the set of all right-infinite words accepted by some finite-state automaton. In our setting, we use the notion of an unrestricted Rabin automaton (see [11], [12], [6]), as well as a related notion of acceptance, in order to provide the analogous characterization of sofic tree shifts.

Let us now illustrate our decidability results. It is easy to decide the emptiness of a sofic tree shift accepted by a given unrestricted Rabin automaton. An idea to decide the surjectivity of a cellular automaton $\tau\colon A^{\Sigma^*} \to A^{\Sigma^*}$ could be to establish the emptiness of the tree language $A^{\Sigma^*} \setminus \tau(A^{\Sigma^*})$. But given a nontrivial tree shift, its complement always fails to be a subshift (nevertheless, Rabin theory guarantees that this tree language is still recognizable by a general Rabin automaton). In order to avoid this obstacle, we introduce the set of full-tree-patterns of a tree shift which is a finite tree language that characterizes the shift. Moreover, the full-tree-patterns of a sofic tree shift are recognizable by a suitably defined finite-tree automaton.

We prove that the recognizable sets of full-tree-patterns form a class which is closed under complementation and for which the emptiness problem is decidable. This allows us to find algorithms establishing both the surjectivity of a cellular automaton $\tau\colon A^{\Sigma^*} \to A^{\Sigma^*}$ and the equality of two sofic-tree shifts presented by unrestricted Rabin automata. With this latter result at hand we can provide a general algorithm establishing the surjectivity of a cellular automaton defined between sofic tree shifts.

In this paper we just detail the decision procedures we mentioned above. The proofs of the results leading to these algorithms are omitted.

2 Definitions and background

In the sequel, we denote by Σ and A two nonempty finite sets. In particular, the set A is called *alphabet* and its elements are called *labels* or *colors*.

2.1 The free monoid Σ^*

For $n \in \mathbb{N}$, we denote by Σ^n the set of all *words* $w = \sigma_1 \sigma_2 \cdots \sigma_n$ of *length* n (where $\sigma_i \in \Sigma$ for $i = 1, 2, \ldots, n$) over Σ. In particular $\varepsilon \in \Sigma^0$ indicates the only word of length 0 called the *empty word*. For $n \geq 1$, we denote by Δ_n the set $\bigcup_{i=0}^{n-1} \Sigma^i$ (that is, the set of all words of length $\leq n-1$).

The *concatenation* of two words $w = \sigma_1 \sigma_2 \cdots \sigma_n \in \Sigma^n$ and $w' = \sigma'_1 \sigma'_2 \cdots \sigma'_m \in \Sigma^m$ is the word $ww' = \sigma_1 \sigma_2 \cdots \sigma_n \sigma'_1 \sigma'_2 \cdots \sigma'_m \in \Sigma^{m+n}$. Then the set $\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n$, equipped with the multiplication given by concatenation, is a monoid with identity element the empty word ε. It is called the *free monoid* over Σ.

From the graph theoretical point of view, Σ^* is the vertex set of the $|\Sigma|$-regular rooted tree. The empty word ε is its root and, for every vertex $w \in \Sigma^*$, the vertices $w\sigma \in \Sigma^*$ (with $\sigma \in \Sigma$) are called the *children* of w. Each vertex is connected by an edge to each of its children.

2.2 Configurations and tree shifts

We denote by A^{Σ^*} the set of all maps $f\colon \Sigma^* \to A$. It is called the *space of configurations* of Σ^* over the alphabet A. When equipped with the *prodiscrete topology* (that is, with the product topology where each factor A of $A^{\Sigma^*} = \prod_{w\in\Sigma^*} A$ is endowed with the discrete topology), the configuration space is a compact, totally disconnected, metrizable space. Also, the free monoid Σ^* has a right action on A^{Σ^*} defined as follows: for every $w \in \Sigma^*$ and $f \in A^{\Sigma^*}$ the configuration $f^w \in A^{\Sigma^*}$ is defined by setting $f^w(w') = f(ww')$ for all $w' \in \Sigma^*$. This action, called the *shift action*, is continuous with respect to the prodiscrete topology.

Recall that a neighborhood basis of a configuration $f \in A^{\Sigma^*}$ is given by the sets $\mathcal{N}(f,n) = \{g \in A^{\Sigma^*} : g|_{\Delta_n} = f|_{\Delta_n}\}$ where $n \geq 1$ (as usual, for $M \subset \Sigma^*$, we denote by $f|_M$ the restriction of f to M).

A subset $X \subset A^{\Sigma^*}$ is called a *subshift* (or *tree shift*, or simply *shift*) provided that X is closed (with respect to the prodiscrete topology) and *shift-invariant* (that is, $f^w \in X$ for all $f \in X$ and $w \in \Sigma^*$).

2.3 Forbidden blocks and shifts of finite type

Let $M \subset \Sigma^*$ be a finite set. A *pattern* is a map $p\colon M \to A$. The set M is called the *support* of p and it is denoted by $\mathrm{supp}(p)$. We denote by A^M the set of all patterns with support M. A *block* is a pattern $p\colon \Delta_n \to A$. The integer n is called the *size* of the block. The set of all blocks is denoted by $\mathcal{B}(A^{\Sigma^*})$.

If X is a subset of A^{Σ^*} and $M \subset \Sigma^*$ is finite, the set of patterns $\{f|_M : f \in X\}$ is denoted by X_M. For $n \geq 1$, the notation X_n is an abbreviation for X_{Δ_n} (that is, the set of all blocks of size n which are restrictions to Δ_n of some configuration in X). We denote by $\mathcal{B}(X)$ the set of all blocks of X (that is, $\mathcal{B}(X) = \bigcup_{n\geq 1} X_n$).

Given a block $p \in \mathcal{B}(A^{\Sigma^*})$ and a configuration $f \in A^{\Sigma^*}$, we say that p *appears* in f if there exists $w \in \Sigma^*$ such that $(f^w)|_{\mathrm{supp}(p)} = p$. If p does not appear in f, we say that f *avoids* p. Let $\mathcal{F}$ be a set of blocks. We denote by $\mathsf{X}(\mathcal{F})$ the set of all configurations in A^{Σ^*} avoiding each block of $\mathcal{F}$, in symbols $\mathsf{X}(\mathcal{F}) = \{f \in A^{\Sigma^*} : (f^w)|_{\Delta_n} \notin \mathcal{F}, \text{ for all } w \in \Sigma^* \text{ and } n \geq 1\}$.

In analogy with the one-dimensional case (see for example [9, Theorem 6.1.21]), we have the following combinatorial characterization of subshifts: *a subset $X \subset A^{\Sigma^*}$ is a subshift if and only if there exists a set $\mathcal{F} \subset \mathcal{B}(A^{\Sigma^*})$ of blocks such that $X = \mathsf{X}(\mathcal{F})$*.

Let $X \subset A^{\Sigma^*}$ be a subshift. A set $\mathcal{F}$ of blocks as above is called a *defining set of forbidden blocks* for X. A subshift is *of finite type* if it admits a finite defining set of forbidden blocks.

Remark 1. We can always suppose that the forbidden blocks of a defining set of a given subshift of finite type all have the same support. This motivates the following definition: a shift of finite type has *memory n* if it admits a defining set of forbidden blocks of size n. Notice that a shift with memory n, also has memory m for all $m \geq n$.

Example 1 (Monochromatic children). Since $\Delta_2 = \{\varepsilon\} \cup \Sigma$, we can identify A^{Δ_2} with $A \times A^{\Sigma}$. Consider the set of blocks

$$\mathcal{F} = \left\{ (a, (a_\sigma)_{\sigma \in \Sigma}) \in A \times A^{\Sigma} : a_\sigma \neq a_{\sigma'} \text{ for some } \sigma, \sigma' \in \Sigma \right\}.$$

The tree shift $\mathsf{X}(\mathcal{F}) \subset A^{\Sigma^*}$ is of finite type and exactly consists of those configurations for which every vertex in Σ^* has monochromatic children. If $|\Sigma| = 2$ and $A = \{0, 1\}$ an example of a configuration in $\mathsf{X}(\mathcal{F})$ is given in Figure 1.
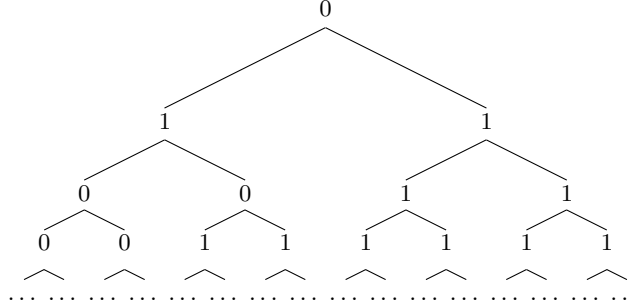
Fig. 1. A configuration of the tree shift presented in Example 1.

2.4 Cellular automata and sofic tree shifts

Let $X \subset A^{\Sigma^*}$ be a tree shift. A map $\tau \colon X \to A^{\Sigma^*}$ is called a *cellular automaton* if it satisfies the following condition: there exists a finite subset $M \subset \Sigma^*$ and a map $\mu \colon A^M \to A$ such that $\tau(f)(w) = \mu((f^w)|_M)$ for all $f \in X$ and $w \in \Sigma^*$. The set M is called a *memory set* for τ and μ is the associated *local defining map*. We assume in the sequel (without loss of generality), that a memory set has the form $M = \Delta_n$, for a suitable $n \geq 1$.

The Curtis-Hedlund-Lyndon theorem gives a topologically characterization of cellular automata: *a map $\tau \colon X \to A^{\Sigma^*}$ is a cellular automaton if and only if it commutes with the shift action (that is, $(\tau(f))^w = \tau(f^w)$ for all $f \in X$ and $w \in \Sigma^*$), and is continuous (with respect to the prodiscrete topology on X).* For a proof in the one-dimensional case, see [9, Theorem 6.2.9]. See also [5, Theorem 1.8.1] and [7], for a more general setting. It immediately follows that the image of a tree shift under a cellular automaton is still a tree shift.

Remark 2. In the definition of a cellular automaton we have assumed that the alphabet of the shift X is the same as the alphabet of its image $\tau(X)$. In this assumption there is no loss of generality because if $\tau \colon X \to B^{\Sigma^*}$, one can always consider X as a subshifts of $(A \cup B)^{\Sigma^*}$. Classically, a cellular automaton is also a selfmapping $\tau \colon X \to X$. By dropping this hypothesis, we deal with a more general notion that, in the one-dimensional case, corresponds to that of *sliding block code* as defined in [9].

A subshift $X \subset A^{\Sigma^*}$ is called *sofic* provided there exist a subshift of finite type $Y \subset A^{\Sigma^*}$ and a cellular automaton $\tau \colon Y \to A^{\Sigma^*}$ such that $X = \tau(Y)$.

Remark 3. Every subshift of finite type is sofic but there are examples of sofic subshifts which are not of finite type (see [9, Example 2.1.5, Example 2.1.9]).

3   Unrestricted Rabin graphs and automata

An *unrestricted Rabin graph*, is a 4-tuple $\mathcal{G} = (S, \Sigma, A, \mathcal{T})$, where S is a nonempty set, called the set of *states* (or *vertices*) of $\mathcal{G}$ and $\mathcal{T}$ is a subset of $S \times A \times S^\Sigma$ whose elements are called *transition bundles*. When the state set S is finite $\mathcal{G} = (S, \Sigma, A, \mathcal{T})$ is called an *unrestricted Rabin automaton*.

Given a transition bundle $t = (s; a; (s_\sigma)_{\sigma \in \Sigma}) \in \mathcal{T}$ we denote by $\mathbf{i}(t) := s \in S$ its *initial state*, by $\lambda(t) := a \in A$ its *label*, by $\mathbf{t}(t) := (s_\sigma)_{\sigma \in \Sigma} \in S^\Sigma$ its *terminal sequence* and by $\mathbf{t}_\sigma(t) := s_\sigma \in S$ its *σ-terminal state*. A *bundle loop on $s \in S$* is a transition bundle $t \in \mathcal{T}$ such that $\mathbf{i}(t) = \mathbf{t}_\sigma(t) = s$ for all $\sigma \in \Sigma$.

An unrestricted Rabin graph $\mathcal{G} = (S, \Sigma, A, \mathcal{T})$ is said to be *essential* provided that for each state $s \in S$ there is a transition bundle starting at s.

Definition 1 (Unrestricted Rabin graph of a configuration). The *unrestricted Rabin graph of a configuration $f \in A^{\Sigma^*}$* is defined by $\mathcal{G}_f = (\Sigma^*, \Sigma, A, \mathcal{T}_f)$ where $\mathcal{T}_f = \{(w; f(w); (w\sigma)_{\sigma \in \Sigma}) : w \in \Sigma^*\}$.

Definition 2 (Homomorphism). A *homomorphism* from $\mathcal{G}_1 = (S_1, \Sigma, A, \mathcal{T}_1)$ to $\mathcal{G}_2 = (S_2, \Sigma, A, \mathcal{T}_2)$ is a map $\alpha \colon S_1 \to S_2$ such that $(\alpha(s); a; (\alpha(s_\sigma))_{\sigma \in \Sigma}) \in \mathcal{T}_2$ for all $(s; a; (s_\sigma)_{\sigma \in \Sigma}) \in \mathcal{T}_1$. By abuse of language/notation, we also denote by $\alpha \colon \mathcal{G}_1 \to \mathcal{G}_2$ such a homomorphism.

Definition 3 (Acceptance). Let $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ be an unrestricted Rabin automaton. We say that a configuration $f \in A^{\Sigma^*}$ is *accepted* (or *recognized*) by $\mathcal{A}$, if there exists a homomorphism $\alpha \colon \mathcal{G}_f \to \mathcal{A}$. In this case, we say that *f is accepted by $\mathcal{A}$ via α*. We denote by $\mathsf{X}_\mathcal{A}$ the set consisting of all those configurations $f \in A^{\Sigma^*}$ accepted by $\mathcal{A}$. An unrestricted Rabin automaton $\mathcal{A}$ is called a *presentation* for $X \subset A^{\Sigma^*}$ provided that $X = \mathsf{X}_\mathcal{A}$.

Remark 4. In the sequel, we shall always consider essential unrestricted Rabin automata. This is not restrictive since, by recursively removing all states that are source of no transition bundles, we can transform any unrestricted Rabin automaton $\mathcal{A}$ into an essential one $\mathcal{A}'$ which accepts the same subset, i.e. such that $\mathsf{X}_\mathcal{A} = \mathsf{X}_{\mathcal{A}'}$.

Remark 5. Explicitly, a configuration $f \in A^{\Sigma^*}$ is accepted by an unrestricted Rabin automaton $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ if there exists a map $\alpha \colon \Sigma^* \to S$ such that $(\alpha(w); f(w); (\alpha(w\sigma))_{\sigma \in \Sigma}) \in \mathcal{T}$ for all $w \in \Sigma^*$.

3.1 Graphical representation

Let $|\Sigma| = k$. We identify Σ with the set $\{0, 1, \ldots, k-1\}$. Hence, a transition bundle of an unrestricted Rabin automaton $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ is a $(k+2)$-tuple $t = (s; a; s_0, \ldots, s_{k-1})$ and it can be visualized as in Figure 2(a). If $|\Sigma| = 2$ and $(s; a; s_0, s_1)$ is a transition bundle, we represent the edge from s to s_0 by a broken line and the edge from s to s_1 by a full line. This makes unnecessary to label the corresponding edges by 0 and 1, respectively (see Figure 2(b)).

(a) A general labeled transition bundle.

(b) A labeled transition bundle of an unrestricted Rabin automaton in which $\Sigma = \{0, 1\}$.
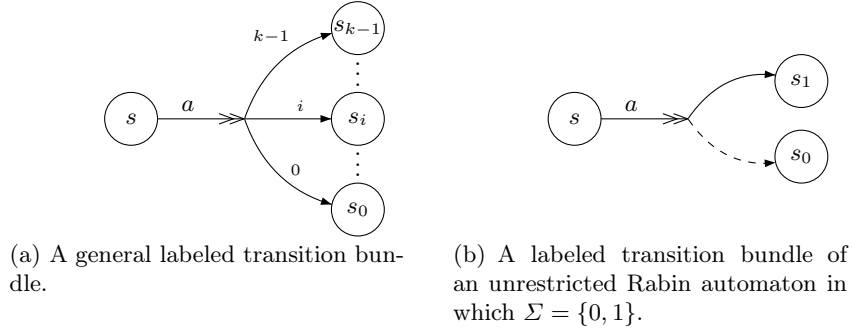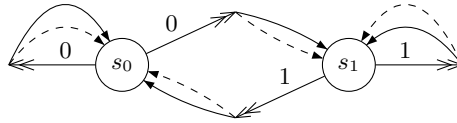
Fig. 2. Representations of a transition bundle.

Example 2. Consider the unrestricted Rabin automaton $\mathcal{A} = (A, \Sigma, A, \mathcal{T})$ where the bundle set is given by

$$\mathcal{T} = \{(a; a; (a_\sigma)_{\sigma \in \Sigma}) \in A \times A \times A^\Sigma : a_\sigma = a_{\sigma'} \text{ for all } \sigma, \sigma' \in \Sigma\}.$$

We then have that $\mathsf{X}_\mathcal{A}$ is the tree shift described in Example 1. If $|\Sigma| = 2$ and $A = \{0, 1\}$ the corresponding automaton is represented in Figure 3.

Fig. 3. The unrestricted Rabin automaton accepting the tree shift of Example 1.

3.2 Unrestricted Rabin automata and sofic shifts

Proposition 1. *Let $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ be an unrestricted Rabin automaton. Then $\mathsf{X}_\mathcal{A}$ is a sofic tree shift. Actually, up to a suitable extension of the alphabet A, there is an effective procedure to construct a tree shift of finite type $Y \subset A^{\Sigma^*}$ and a cellular automaton $\tau \colon Y \to A^{\Sigma^*}$ such that $\mathsf{X}_\mathcal{A} = \tau(Y)$.*

Let $M \subset \Sigma^*$ be a nonempty subset and $p \in A^M$ a pattern with support M. Given a word $w \in \Sigma^*$ we set $wM = \{wm : m \in M\} \subset \Sigma^*$ and denote by ${}^w p \in A^{wM}$ the pattern with support wM defined by $({}^w p)(wm) = p(m)$ for all $m \in M$.

Definition 4 (Unrestricted Rabin automaton associated with a cellular automaton). Let $X \subset A^{\Sigma^*}$ be a tree shift of finite type and let $\tau \colon X \to A^{\Sigma^*}$ be a cellular automaton. Let $M = \Delta_n \subset \Sigma^*$ be a memory set for τ such that $n \geq 2$ and denote by $\mu \colon A^M \to A$ the corresponding local defining map. Fix $M' = \Delta_{n-1}$. The *unrestricted Rabin automaton $\mathcal{A}(\tau, M, X)$ associated with τ* is defined by $\mathcal{A}(\tau, M, X) = (X_{M'}, \Sigma, A, \mathcal{T})$, where $\mathcal{T} \subset X_{M'} \times A \times (X_{M'})^{\Sigma}$ consists of the bundles $(p; b; (p_\sigma)_{\sigma \in \Sigma})$ such that (*i.*) $p|_{\sigma M' \cap M'}$ equals $({}^\sigma p_\sigma)|_{\sigma M' \cap M'}$ for all $\sigma \in \Sigma$ (that is, $p(\sigma m) = p_\sigma(m)$ whenever $\sigma m \in \sigma M' \cap M'$); (*ii.*) the block $\bar{p} \colon M \to A$ coinciding with p on M' and with ${}^\sigma p_\sigma$ on $\sigma M'$ belongs to X_M for all $\sigma \in \Sigma$ (such a block $\bar{p} \in X_M$ is denoted by $\overline{(p; (p_\sigma)_{\sigma \in \Sigma})}$); (*iii.*) $b = \mu\left(\overline{(p; (p_\sigma)_{\sigma \in \Sigma})}\right)$.
A transition bundle of $\mathcal{A}(\tau, M, X)$ is illustrated in Figure 4 for $|\Sigma| = 2$.
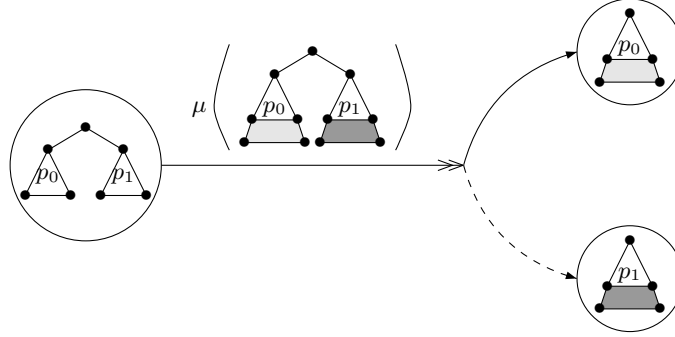
Fig. 4. A transition bundle of $\mathcal{A}(\tau, M, X)$ when $|\Sigma| = 2$.

Proposition 2. *Let $X \subset A^{\Sigma^*}$ be a tree shift of finite type with memory $n-1$ and $\tau \colon X \to A^{\Sigma^*}$ be a cellular automaton with memory set Δ_n. Then $\mathsf{X}_{\mathcal{A}(\tau, \Delta_n, X)} = \tau(X)$.*

Remark 6. Conditions on Δ_n in Proposition 2 are not restrictive. Proposition 2 says that $\mathcal{A}(\tau, \Delta_n, X)$ is a presentation of $\tau(X)$. In fact, we can actually show how to construct a pre-image of a configuration in $\mathsf{X}_{\mathcal{A}(\tau, \Delta_n, X)}$. This leads in particular to a presentation of X as well.

Proposition 1 and Proposition 2 imply the following result. The bottom-up version of it has been proved by Béal and Aubrun in [4].

Corollary 1. *A tree shift $X \subset A^{\Sigma^*}$ is sofic if and only if it is accepted by some unrestricted Rabin automaton.*

3.3 Deterministic and co-deterministic presentations

An unrestricted Rabin automaton $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ is *deterministic* if, for each state $s \in S$, the transition bundles starting at s carry different labels. Analogously, $\mathcal{A}$ is *co-deterministic* if, for each sequence $\mathbf{s} \in S^{\Sigma}$, the transition bundles terminating at $\mathbf{s}$ (if there are any) carry different labels.

As stated below, for each unrestricted Rabin automaton $\mathcal{A}$ there exists a co-deterministic unrestricted Rabin automaton accepting the same shift.

Theorem 1 (Subset construction). *Let $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ be an unrestricted Rabin automaton. There exists a co-deterministic unrestricted Rabin automaton $\mathcal{A}_{\mathrm{cod}}$ such that $\mathsf{X}_{\mathcal{A}} = \mathsf{X}_{\mathcal{A}_{\mathrm{cod}}}$.*

The statement of the above theorem fails to hold, in general, for deterministic unrestricted Rabin automata, as shown in the following counterexample.

Example 3 (A sofic shift not admitting a deterministic presentation). Consider the tree shift X presented in Example 1. A non-deterministic presentation of X is given in Example 2. Suppose that X admits a deterministic presentation $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$. First observe that, in this case, each *accessible* state (that is, each state that can be reached by a transition bundle), admits exactly one transition bundle starting at it. Thus for every accessible state $s \in S$ there exists exactly one configuration $f_s \in X$ accepted by a homomorphism $\alpha_s \colon \Sigma^* \to S$ *starting* at s, that is, such that $\alpha_s(\varepsilon) = s$. This implies that any state determines at most $|A|$ configurations (indeed, for a state s that is not accessible, there are at most $|A|$ bundles that start at s and all of these bundles end in accessible states). Therefore $\mathcal{A}$ accepts only finitely many different configurations, which contradicts the fact that X is infinite.

4 Full-tree-patterns and finite-tree automata

Recall that a *k-ary rooted tree* is a rooted tree in which each vertex has at most k children. A *leaf* is a vertex without children. A *full k-ary rooted tree* is a rooted tree in which every vertex other than the leaves has k children. Hence Σ^* is the full k-ary rooted tree with no leaves, where $k = |\Sigma|$. A *subtree* of Σ^* is a connected subgraph of Σ^*. We shall always suppose that a subtree of Σ^* contains the root ε. If $T \subset \Sigma^*$ is a subtree and $w \in T$, we denote by $\Sigma_T(w)$ the set $\{\sigma \in \Sigma : w\sigma \in T\}$. Hence $w \in T$ is a leaf if and only if $\Sigma_T(w) = \varnothing$.

Given a subtree T, we denote by T^+ the subtree $T \cup \{w\sigma : w \in T, \sigma \in \Sigma\}$. Notice that T^+ is always a full subtree. If T is a full subtree, then T^+ is obtained by adding all the k children of each leaf in T.

Notice that for each $n \geq 1$ the set Δ_n is a full subtree whose leaves are the elements in Σ^{n-1}. Moreover, $\Delta_n^+ = \Delta_{n+1}$.

Finite full subtrees correspond to finite and complete prefix codes in [2].

A pattern defined on a finite full subtree T is called *full-tree-pattern*. The set of all full-tree-patterns is denoted by $\mathfrak{T}(A^{\Sigma^*})$. Given a shift $X \subset A^{\Sigma^*}$, we denote by $\mathfrak{T}(X)$ the set of all full-tree-patterns of X (that is, $\mathfrak{T}(X) = \bigcup_{T \subset \Sigma^*} X_T$, where the union ranges over all finite full subtrees T of Σ^*).

Definition 5 (Sub-bundle). Let $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ be an unrestricted Rabin automaton. Let $M \subset \Sigma$ be a subset. A tuple $(s; a; (s_\sigma)_{\sigma \in M}) \in S \times A \times S^M$ is called a *sub-bundle* of a transition bundle $(\bar{s}; \bar{a}; (\bar{s}_\sigma)_{\sigma \in \Sigma}) \in \mathcal{T}$ provided $s = \bar{s}$, $a = \bar{a}$, and $s_\sigma = \bar{s}_\sigma$ for each $\sigma \in M$.

Definition 6. Let $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ be an unrestricted Rabin automaton. Let $T \subset \Sigma^*$ be a subtree and let $f \colon T \to A$ be a map. One says that f is *accepted by* $\mathcal{A}$ if there exists a map $\alpha \colon T \to S$ such that, for each $w \in T$, $(\alpha(w); f(w); (\alpha(w\sigma))_{\sigma \in \Sigma_T(w)})$ is a sub-bundle of some $t \in \mathcal{T}$. In this case we say that f is accepted by $\mathcal{A}$ *via* α.

Note that, for a leaf $w \in T$, this latter acceptance condition reduces to saying that there exists a transition bundle starting at $\alpha(w)$ with label $f(w)$ (in fact, α is not defined on $w\sigma$ for any $\sigma \in \Sigma$).

Proposition 3. *Let* $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ *be an unrestricted Rabin automaton. Let* $T \subset \Sigma^*$ *be a subtree and suppose that* $f \in A^T$ *is accepted by* $\mathcal{A}$*. Then there exists a configuration* $\bar{f} \in \mathsf{X}_\mathcal{A}$ *such that* $f = \bar{f}|_T$*.*

We have the following characterization of acceptance which immediately results from Definition 6.

Proposition 4. *Let* $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ *be an unrestricted Rabin automaton. Let* $T \subset \Sigma^*$ *be a finite full subtree. A full-tree-pattern* $p \in A^T$ *is accepted by* $\mathcal{A}$ *if and only if there exists a map* $\alpha \colon T^+ \to S$ *such that* $(\alpha(w); p(w); (\alpha(w\sigma))_{\sigma \in \Sigma}) \in \mathcal{T}$ *for each* $w \in T$*.*

By abuse of language, if this acceptance condition holds and there is no ambiguity, we say that the full-tree-pattern p *is accepted by* $\mathcal{A}$ *via* α. Obviously, Proposition 4 applies whenever $T = \Delta_n$ for some $n \geq 1$ (recall that in this case $T^+ = \Delta_{n+1}$).

The following result follows from Proposition 3.

Corollary 2. *Let* $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ *be an unrestricted Rabin automaton. Let* $p \in \mathfrak{T}(A^{\Sigma^*})$ *be a full-tree-pattern. Then* $p \in \mathfrak{T}(\mathsf{X}_\mathcal{A})$ *if and only if* p *is accepted by* $\mathcal{A}$*.*

Remark 7. The blocks of a subshift determine the subshift. In fact, given two subshifts $X, Y \subset A^{\Sigma^*}$, we have $X = \mathsf{X}(\mathcal{B}(A^{\Sigma^*}) \setminus \mathcal{B}(X))$ so that $X = Y$ if and only if $\mathcal{B}(X) = \mathcal{B}(Y)$. This fact obviously generalizes to full-tree-patters: $X = Y$ if and only if $\mathfrak{T}(X) = \mathfrak{T}(Y)$.

4.1 Finite-tree automata

A *finite-tree automaton* is an unrestricted Rabin automaton $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ for which a subset $\mathcal{I} \subset S$ of *initial states* and a state $F \in S$, called *final state*, are specified. We shall denote it by $\mathcal{A}(\mathcal{I}, F)$. We say that a full-tree-pattern $p \in A^T$ is *accepted by* $\mathcal{A}(\mathcal{I}, F)$ if there exists a map $\alpha \colon T^+ \to S$ such that (*i.*) p is accepted

by $\mathcal{A}$ via α (see Proposition 4); (*ii.*) $\alpha(\varepsilon) \in \mathcal{I}$; (*iii.*) $\alpha(w) = F$ if $w \in T^+ \setminus T$. We denote by $\mathfrak{T}(\mathcal{A}(\mathcal{I}, F))$ the set of all full-tree-patterns accepted by $\mathcal{A}(\mathcal{I}, F)$. A set of full-tree-patterns is called *recognizable* if it is of the form $\mathfrak{T}(\mathcal{A}(\mathcal{I}, F))$, for some finite-tree automaton $\mathcal{A}(\mathcal{I}, F)$. A finite-tree automaton $\mathcal{A}(\mathcal{I}, F)$ is *co-deterministic* if the unrestricted Rabin automaton $\mathcal{A}$ is co-deterministic.

Remark 8. As explained in Section 4, we only consider essential unrestricted Rabin automata. As far as finite-tree automata are concerned, we relax this assumption: each non final state is the source of some transition bundle, but no condition is required for the final state.

An unrestricted Rabin automaton $\mathcal{A} = (S, \Sigma, A, \mathcal{T})$ is called *co-complete* if for each $\mathbf{s} \in S^\Sigma$ and $a \in A$, there exists a transition bundle in $\mathcal{T}$ labeled by a and ending at $\mathbf{s}$. A finite-tree automaton $\mathcal{A}(\mathcal{I}, F)$ is *co-complete* if the unrestricted Rabin automaton $\mathcal{A}$ is co-complete.

A slight adaptation in the proof of Theorem 1 leads to the following result.

Theorem 2. *Let $\mathcal{A}$ be an unrestricted Rabin automaton. Then there is an effective procedure to construct a co-deterministic finite-tree automaton $\mathcal{A}_{\mathrm{cod}}(\mathcal{I}, F)$ such that $\mathfrak{T}(\mathsf{X}_\mathcal{A}) = \mathfrak{T}(\mathcal{A}_{\mathrm{cod}}(\mathcal{I}, F))$.*

The recognizable sets of full-tree-patterns form a class which is closed under complementation, as stated in the following theorem.

Theorem 3. *Let $\mathcal{A}(\mathcal{I}, F)$ be a co-deterministic finite-tree automaton. Then there exists a co-complete and co-deterministic finite-tree automaton $\mathcal{A}_{\complement}(\mathcal{I}_\complement, F_\complement)$ such that $\mathfrak{T}(A^{\Sigma^*}) \setminus \mathfrak{T}(\mathcal{A}(\mathcal{I}, F)) = \mathfrak{T}(\mathcal{A}_\complement(\mathcal{I}_\complement, F_\complement))$.*

Corollary 3. *Let $\mathcal{A}$ be an unrestricted Rabin automaton. Then there is an effective procedure to construct a co-complete and co-deterministic finite-tree automaton $\mathcal{A}_\complement(I, F)$ (with a single initial state) which accepts the complement of the set of all full-tree-patterns accepted by $\mathcal{A}$, in formulæ, $\mathfrak{T}(\mathcal{A}_\complement(I, F)) = \mathfrak{T}(A^{\Sigma^*}) \setminus \mathfrak{T}(\mathsf{X}_\mathcal{A})$.*

Corollary 4. *Let $\mathcal{A}$ be an unrestricted Rabin automaton. Let $\mathcal{A}_\complement(I, F)$ be as in Corollary 3. Then $\mathsf{X}_\mathcal{A} = A^{\Sigma^*}$ if and only if $\mathfrak{T}(\mathcal{A}_\complement(I, F)) = \varnothing$.*

The emptiness problem for finite-tree automata The emptiness problem for an unrestricted Rabin automaton is trivial (every nonempty essential automaton accepts at least a configuration), but this argument does not apply to the case of finite-tree automata. In this section we present an effective procedure to establish the emptiness of recognizable set of full-tree-patterns.

First, we define the *height of a finite subtree* $T \subset \Sigma^*$ as the minimal $n \in \mathbb{N}$ such that $T \subset \Delta_n$. The *height of a full-tree-pattern* $p \in A^T$ is the height of the (finite full) subtree T.

Let $\mathcal{A}(\mathcal{I}, F)$ be a finite-tree-automaton and let us show that there is an algorithm which establishes whether or not $\mathfrak{T}(\mathcal{A}(\mathcal{I}, F)) = \varnothing$. Observe that $\mathfrak{T}(\mathcal{A}(\mathcal{I}, F))$ is nonempty if and only if it contains a pattern of height $\leq |S|$,

where S is the state set of $\mathcal{A}$ (we do not prove this fact in detail). Since there are finitely many full-tree-patterns of height $\leq |S|$ one can effectively check whether or not they are accepted by $\mathcal{A}(\mathcal{I}, F)$.

Since in principle we have to check all possible maps $\alpha\colon \Delta_{|S|+1} \to S$, this algorithm has exponential complexity in the size of S.

An algorithm establishing whether two sofic shifts coincide The *join* of $\mathcal{A}_1 = (S_1, \Sigma, A, \mathcal{T}_1)$ and $\mathcal{A}_2 = (S_2, \Sigma, A, \mathcal{T}_2)$ is the unrestricted Rabin automaton $\mathcal{A}_1 * \mathcal{A}_2 = (S_1 \times S_2, \Sigma, A, \mathcal{T}_\times)$ where $((s_1, s_2); a; (s'_\sigma, s''_\sigma)_{\sigma \in \Sigma}) \in \mathcal{T}_\times$ if and only if $(s_1; a; (s'_\sigma)_{\sigma \in \Sigma}) \in \mathcal{T}_1$ and $(s_2; a; (s''_\sigma)_{\sigma \in \Sigma}) \in \mathcal{T}_2$. Notice that $\mathsf{X}_{\mathcal{A}_1 * \mathcal{A}_2} = \mathsf{X}_{\mathcal{A}_1} \cap \mathsf{X}_{\mathcal{A}_2}$. Moreover, $\mathcal{A}_1 * \mathcal{A}_2$ is co-complete (respectively, co-deterministic), if $\mathcal{A}_1$ and $\mathcal{A}_2$ are co-complete (resp., co-deterministic).

We are now in position to describe our algorithm: let $\mathcal{A}_1 = (S_1, \Sigma, A, \mathcal{T}_1)$ and $\mathcal{A}_2 = (S_2, \Sigma, A, \mathcal{T}_2)$ be two unrestricted Rabin automata. Note that, by Remark 7, it suffices to establish whether or not

$$\mathfrak{T}(\mathsf{X}_{\mathcal{A}_1}) \setminus \mathfrak{T}(\mathsf{X}_{\mathcal{A}_2}) = \varnothing = \mathfrak{T}(\mathsf{X}_{\mathcal{A}_2}) \setminus \mathfrak{T}(\mathsf{X}_{\mathcal{A}_1}). \tag{1}$$

First construct the co-complete and co-deterministic finite-tree automata $\mathcal{A}'_1(I_1, F_1)$ and $\mathcal{A}'_2(I_2, F_2)$ as in Corollary 3, associated with $\mathcal{A}_1$ and $\mathcal{A}_2$, respectively. Consider the finite-tree automaton $(\mathcal{A}'_1 * \mathcal{A}'_2)(\mathcal{I}_1, F)$, where $\mathcal{I}_1 = (S_1 \setminus \{I_1\}) \times \{I_2\}$ and $F = (F_1, F_2)$. It can be seen that $\mathfrak{T}((\mathcal{A}'_1 * \mathcal{A}'_2)(\mathcal{I}_1, F)) = \mathfrak{T}(\mathsf{X}_{\mathcal{A}_1}) \setminus \mathfrak{T}(\mathsf{X}_{\mathcal{A}_2})$. Analogously, by defining $\mathcal{I}_2 = \{I_1\} \times (S_1 \setminus \{I_2\})$ one has $\mathfrak{T}((\mathcal{A}'_1 * \mathcal{A}'_2)(\mathcal{I}_2, F)) = \mathfrak{T}(\mathsf{X}_{\mathcal{A}_2}) \setminus \mathfrak{T}(\mathsf{X}_{\mathcal{A}_1})$.

Thus (1) holds if and only if $\mathfrak{T}((\mathcal{A}'_1 * \mathcal{A}'_2)(\mathcal{I}_1, F)) \bigcup \mathfrak{T}((\mathcal{A}'_1 * \mathcal{A}'_2)(\mathcal{I}_2, F)) = \varnothing$. An effective procedure to establish this latter equality is then provided by the solution to the emptiness problem.

Remark 9. The algorithm above has exponential complexity in the maximal size of the state sets of the unrestricted Rabin automata. A different procedure can be applied to the class of *irreducible* unrestricted Rabin automata by using a minimization process. Actually, in [3] it is shown that there exists a canonical minimal co-deterministic presentation of an irreducible sofic tree shift. Thus another possible decision algorithm consists in computing the minimal presentations of the two shifts and checking whether they coincide or not. In this case Theorem 1 is needed while the procedure for the emptiness problem is not required. Hence this algorithm has in general an exponential complexity. The complexity can be reduced to be polynomial by only considering the class of co-deterministic irreducible tree shifts.

An algorithm establishing the surjectivity of cellular automata Observe first that giving a sofic shift $X \subset A^{\Sigma^*}$ corresponds, equivalently, to giving a shift of finite type $Z \subset A^{\Sigma^*}$ and a surjective cellular automaton $\tau'\colon Z \to X$, or an unrestricted Rabin automaton $\mathcal{A}$ such that $X = \mathsf{X}_{\mathcal{A}}$. Propositions 1 and 2 provide an effective procedure to switch from one representation to the other.

Let $X, Y \subset A^{\Sigma^*}$ be two sofic shifts and $\tau \colon X \to Y$ a cellular automaton. Let us show that it is decidable whether τ is surjective or not. Let $Z \subset A^{\Sigma^*}$ and $\tau' \colon Z \to X$ as above. Now the cellular automaton $\tau \colon X \to Y$ is surjective if and only if the composite cellular automaton $\tau \circ \tau' \colon Z \to Y$ is surjective. Let $n \in \mathbb{N}$ be large enough so that the cellular automaton $\tau \circ \tau'$ has memory set Δ_n and that $n - 1$ is the memory of Z. By Proposition 2, the unrestricted Rabin automaton $\mathcal{A}(\tau \circ \tau', \Delta_n, Z)$ having state set Z_{n-1} is a presentation of $\tau(X)$. Then, it suffices to apply the algorithm in previous section to establish whether $Y = \tau(X)$.

Remark 10. If $X = Y = A^{\Sigma^*}$, then the algorithm becomes much simpler. Indeed, it can be proved by virtue of Corollary 4, Corollary 3 and by using the emptiness algorithm.

References

1. S. Amoroso and Y. N. Patt, *Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures*, J. Comput. System Sci., 6 (1972), pp. 448–464.
2. N. Aubrun, *Dynamique symbolique des systèmes 2D et des arbres infinis*, PhD thesis, Université Paris-Est, 2011.
3. N. Aubrun and M.-P. Béal, *Sofic and almost of finite type tree-shifts*, in CSR 2010, vol. 6072 of Lecture Notes in Comput. Sci., Springer, Berlin, 2010, pp. 12–24.
4. ———, *Sofic tree-shifts*. To appear, 2012.
5. T. Ceccherini-Silberstein and M. Coornaert, *Cellular Automata and Groups*, Springer Monographs in Mathematics. Berlin, 2010.
6. H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi, *Tree automata techniques and applications.* Available on: `http://www.grappa.univ-lille3.fr/tata`, 2007.
7. F. Fiorenzi, *Cellular automata and strongly irreducible shifts of finite type*, Theoret. Comput. Sci., 299 (2003), pp. 477–493.
8. J. Kari, *Reversibility and surjectivity problems of cellular automata*, J. Comput. System Sci., 48 (1994), pp. 149–182.
9. D. A. Lind and B. H. Marcus, *An Introduction to Symbolic Dynamics and Coding*, Cambridge University Press, Cambridge, 1995.
10. D. E. Muller and P. E. Schupp, *The theory of ends, pushdown automata, and second-order logic*, Theoret. Comput. Sci., 37 (1985), pp. 51–75.
11. M. O. Rabin, *Decidability of second-order theories and automata on infinite trees*, Trans. Am. Math. Soc., 141 (1969), pp. 1–35.
12. J. W. Thatcher and J. B. Wright, *Generalized finite automata theory with an application to a decision problem of second-order logic*, Math. Syst. Theory, 2 (1968), pp. 57–81.